REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 21-04-2006	**2. REPORT TYPE** Conference Paper POSTPRINT	**3. DATES COVERED** *(From - To)* 2005 - 2006

4. TITLE AND SUBTITLE

Modeling and Simulation of Satellite Subsystems for End-to-End Spacecraft Modeling

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

William K. Schum, Christina M. Doolittle, and George A. Boyarko

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
AND ADDRESS(ES)

Air Force Research Laboratory
Space Vehicles
3550 Aberdeen Ave SE
Kirtland AFB, NM 87117-5776

8. PERFORMING ORGANIZATION REPORT NUMBER

AFRL-VS-PS-TP-2006-1027

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited. (Clearance #VS06-0136)

13. SUPPLEMENTARY NOTES

Published in the Proceedings of SPIE: Modeling and Simulation for Military Applications, Vol 6228, 21 Apr 06, Paper 622804-1.

14. ABSTRACT

During the past ten years, the Air Force Research Laboratory (AFRL) has been simultaneously developing high-fidelity spacecraft payload models as well as a robust distributed simulation environment for modeling spacecraft subsystems. Much of this research has occurred in the Distributed Architecture Simulation Laboratory (DASL). AFRL developers working in the DASL have effectively combined satellite power, attitude pointing, and communication link analysis subsystem models with robust satellite sensor models to create a first-order end-to-end satellite simulation capability. The merging of these two simulation areas has advanced the field of spacecraft simulation, design, and analysis, and enabled more in-depth mission and satellite utility analyses. A core capability of the DASL is the support of a variety of modeling and analysis efforts, ranging from physics and engineering-level modeling to mission and campaign-level analysis. The flexibility and agility of this simulation architecture will be used to support space mission analysis, military utility analysis, and various integrated exercises with other military and space organizations via direct integration, or through DOD standards such as Distributed Interaction Simulation. This paper discusses the results and lessons learned in modeling satellite communication link analysis, power, and attitude control subsystems for an end-to-end satellite simulation. It also discusses how these spacecraft subsystem simulations feed into and support military utility and space mission analyses.

15. SUBJECT TERMS

Space Vehicles; Spacecraft Modeling; Satellite Subsystems, Satellite Simulation; Distributed Architecture Simulation Laboratory; DASL; End-to-End Satellite Simulation; Space Mission Analysis

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** William K. Schum
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	Unlimited	11	**19b. TELEPHONE NUMBER** *(include area code)* 505-846-4478

Modeling and Simulation of Satellite Subsystems for End-to-End Spacecraft Modeling

William K. Schum[§], Christina M. Doolittle[*], and George A. Boyarko[*]

During the past ten years, the Air Force Research Laboratory (AFRL) has been simultaneously developing high-fidelity spacecraft payload models as well as a robust distributed simulation environment for modeling spacecraft subsystems. Much of this research has occurred in the Distributed Architecture Simulation Laboratory (DASL). AFRL developers working in the DASL have effectively combined satellite power, attitude pointing, and communication link analysis subsystem models with robust satellite sensor models to create a first-order end-to-end satellite simulation capability. The merging of these two simulation areas has advanced the field of spacecraft simulation, design, and analysis, and enabled more in-depth mission and satellite utility analyses. A core capability of the DASL is the support of a variety of modeling and analysis efforts, ranging from physics and engineering-level modeling to mission and campaign-level analysis. The flexibility and agility of this simulation architecture will be used to support space mission analysis, military utility analysis, and various integrated exercises with other military and space organizations via direct integration, or through DOD standards such as Distributed Interaction Simulation. This paper discusses the results and lessons learned in modeling satellite communication link analysis, power, and attitude control subsystems for an end-to-end satellite simulation. It also discusses how these spacecraft subsystem simulations feed into and support military utility and space mission analyses.

1.0 INTRODUCTION

Designing, building, and launching spacecraft is an expensive and risky endeavor. The use of modeling and simulation tools can reduce the cost and risk of this process by providing an environment for performing engineering trade studies, aiding the discovery of behavioral modes and characteristics, and conducting space mission planning.

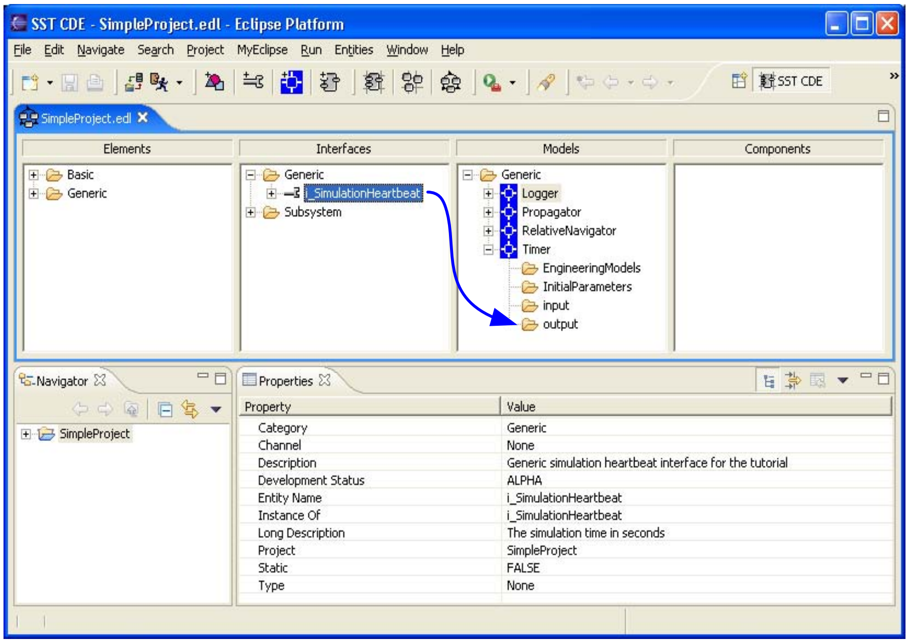
The Distributed Architecture Simulation Laboratory[1] (DASL) is a simulation environment hosted by the AFRL Space Vehicles Directorate at Kirtland Air Force Base in NM. The DASL is a state-of-the-art facility that utilizes Linux, Solaris, Windows and real-time operating systems on powerful workstations networked on a Gigabit network. Simulation modeling in the DASL is done in a modular fashion that lends itself to code reuse.

Code for satellite subsystem component models is written using Matlab and C++ by domain experts who have a strong understanding of the subsystem being modeled. These models are then integrated into system simulations using the System Simulation Toolkit[2] (SST). The SST is a simulation development environment created by Photon Research Associates under several phases of Small

[*] Research Aerospace Engineer, U.S. Air Force Research Laboratory, Kirtland AFB, NM

[§] Lead, Modeling and Simulation Development, U.S. Air Force Research Laboratory, Kirtland AFB, NM

Modeling and Simulation for Military Applications, edited by Kevin Schum, Alex F. Sisti
Proc. of SPIE Vol. 6228, 622804, (2006) · 0277-786X/06/$15 · doi: 10.1117/12.665745



Property	Value
Category	Generic
Channel	None
Description	Generic simulation heartbeat interface for the tutorial
Development Status	ALPHA
Entity Name	i_SimulationHeartbeat
Instance Of	i_SimulationHeartbeat
Long Description	The simulation time in seconds
Project	SimpleProject
Static	FALSE
Type	None

require a fraction of the calculations. If both models are expected to complete their calculations within a very short simulation time step, and they both are not complete when the simulation is ready to step forward, the simulation may stall. The SST lends itself to controlling the various processing threads and coordinating their execution.

The architecture for this simulation utilized a distributed processor design. This allows various components to be run on different operating systems, faster computers, or geographically separated networks. The modular architecture allows any one of the key components to be replaced without rebuilding the entire simulation. The goal of this approach to satellite modeling is to keep the simulation generic and not specific to a given bus design. The flight software tends to be very bus specific, so that portion of the simulation generally needs to be replaced with each major simulation. For this research, a form of surrogate flight software was embedded in the primary simulation controller. The surrogate provided some greatly simplified flight control rules, thereby mitigating the need for a full closed-loop[4] attitude determination and control system (ADCS) model.

This simulation is intended to model the power utilization, attitude pointing, and communication link (IPAC) as a system, and capture data from each component at each time step throughout the simulation. To model the IPAC components, an environment in which they operate had to be simulated. The orbit propagation for this simulation was done using STK. This allowed SST to step the satellite forward in its orbit, providing the satellite's new position, velocity, and associated errors to the other simulation components at each time-step.

Data for the simulation is recorded on each time step in a table, and is indexed by time step. The SST allows for virtually any variable to be identified for capture in the data table. Post-simulation analyses can then be run to assess the performance of each component. This will be discussed in the results section.

Payload Simulation

The goal of the payload model in this simulation environment was to represent how the payload interacted with the IPAC components through the SST architecture. To that end, this simulation utilized a generic sensor model that took into account the orbital trajectory, altitude, and relative position, velocity, and orientation of the satellite. The orientation, position of the sun, and eclipse data were provided on each time step by the Attitude model. On each time step, the payload returned its power consumption and a realistic amount of data back to the simulation. This data was passed to the other components so the power and communications could be modeled.

Power Subsystem

The power subsystem model is written in C++. It determines power generation, power usage and storage, and determines the new state of charge of the battery. During initialization of the power model, characteristics of the power subsystem, such as power source and storage capabilities, are chosen. At every time step of the simulation, power usage information is delivered from the payload model and any other subsystem models being used in the sim. Eclipse and attitude data are also delivered to the power model.

The power subsystem model first calculates power generation for the solar arrays. Eclipse conditions, as well as the solar incidence angle to the arrays, are taken into account for these calculations. Power being used by the payload and other subsystems, as well as a steady power drain from command and data handling, parasitic loads, and other critical systems is then used to calculate the net power available. If the spacecraft is not in eclipse, and there is ample power to feed the systems that need it, any excess power is then used to charge the battery. A negative net power means that the battery is being discharged, whether by an eclipse condition or simply by a power usage in excess of that generated by the solar panels, and the battery charge will decrease accordingly.

The power model is a relatively simple, low fidelity model. It does not account for variable battery charging rates, battery degradation, line losses, or any other efficiency issues. Its purpose is more to give a first-order estimate of the power status at any time, given demands upon the system, quickly enough that the calculations can be completed in a time step (which may be a fraction of a second), leaving time for message passing before the start of the next time step. Additional fidelity such as a power management system, load shedding, and line losses are planned for the next iteration of the model.

Attitude Class

The attitude class models the essential characteristics of the attitude control system rather than the attitude dynamics of the spacecraft. The ADCS model is a C++ class consisting of functions that can take a series of “ADCS Mode” commands (such as ‘nadir alignment’ and ‘hold solar inertial’) and return the resulting attitude information and pointing kinematics. The model provides structure and solar panel pointing information in direction cosine matrix or quaternion form, as well as reaction wheel power requirements for a particular maneuver. The ADCS only calculates information at a specified time and passes it to SST; no future or past information is stored within the class. ADCS calculates the true quaternion as well as a quaternion subjected to Gaussian noise whose parameters are set at the scenario start by the user. The simplicity of the ADCS architecture provides for a plug-and-play functionality that can work with a wide variety of different drivers.

The ADCS model is called every time step through the use of an update function. The update function passes the current time in Julian Date format as well as state data at that specific time. The ADCS then calculates and stores pointing information for that time step only. These values can then be accessed by SST for use in other calculations by other models or logging. Inputs of pointing mode, target location and pointing error statistics are set by a variety of calls to set specific variables before the update function is called. Satellite parameters such as the inertia matrix and maximum slew rate are set in the initialization of the attitude class and read in before the scenario start.

Another role the ADCS class fills is to aid mission planning and power usage. The ADCS has specific functions that will calculate the time needed to slew from one pointing mode to another. The function calculates the magnitude of the slew angle at each time step and then searches through all slew solutions with knowledge of the maximum slew rate constraint, which is set in the initialization of the attitude class. The slew time fitting the constraints is returned to SST for use in the scheduling of maneuvers. Power needed to slew is similar in the sense that a difference quaternion is calculated from two pointing modes at specific times. The slew rate and torques needed to perform the slew are calculated and fed into a reaction wheel model which will return the power requirements for the maneuver. In the case of calculating slew time, it needs ephemeris (state time, position and velocity) information from SST at the future times up to the amount of time the user would like to be considered for calculating the end slew time. In the case where it calculates the slew power, it needs a final slew orientation and the specific time in Julian Date when the slew should end (in this case, slew time is an input). During all slew and power time calculations the attitude model calculates the rotation of the earth during slew and the location on the WGS84 ellipsoid[5] during the maneuver.

Communication Subsystem Model

The spacecraft communication subsystem model estimates the maximum amount of data transmitted from the spacecraft back to the ground, given information on spacecraft and ground station antennas and the spacecraft's location. The communication model begins a simulation with data about the spacecraft's antenna and a list of user-specified ground stations of interest. Data for site antennas is included in a flat file to be read as needed. The model also contains a data queue, depicting the amount of data waiting to be down-linked, which may be increased at any time step by the sim controller, and which is decremented at every time step with a successful transmission.

At each time step of the sim, the model receives its state: on, off, or transmit. For an 'off' state, it takes no action; for 'on', it returns information on its minimal (warm-up) power drain to be delivered to the power subsystem. When the model is tasked to transmit, it collects the state vector provided to it and begins to check the ground station selections provided by the user. It will check every selected ground station for line-of-sight; if access is available for that station, it feeds data on range to the ground station and other parameters to the link equation[6] function, which returns the data rate to that station. Data rates for each of the ground stations of interest are sorted as they come in, and the model then directs, or redirects, the transmission to the ground station with the best link.

The model will assume that the transmission to the ground will last for the duration of the time step. It determines the amount of data transmitted based on the calculated data rate, which cannot exceed the amount of data in the buffer. The model then reports which ground station was contacted, how much data was sent to that station, the current total data amount downloaded for the entire simulation, and how much data is left in the data queue. Power consumption is also reported for the 'transmit' state. The model will accommodate a variable time step, but since the assumption is made that no parameters change during the time step, the step used needs to be short enough to not compromise efficiency.

Verification, Validation, and Configuration Control

One of the first questions asked by end users of simulations is about validation. "Why should I trust the results of your simulation?" is a very common question. Also, with multiple developers working on the simulation, control of various software configurations impacts the tool's credibility.

The process of verification, validation, and accreditation (VV&A) involves verifying that the tool was coded correctly, validating the results against known models or hand generated solutions, and accrediting the tool through some configuration control protocol. The process utilized by the AFRL Simulation and Technology Assessment branch is to identify the requirements of the simulation, develop the appropriate algorithms, code and unit test them, integrate them into the simulation, and perform system testing of the overall simulation. Verification of the code is done by tracking each variable as it moves among the simulation modules across the SST backbone. Validation of the overall simulation is done by a panel of reviewers consisting of systems and astronautical engineers, computer scientists, a product manager, and a principal investigator.

Once the simulation reaches a reasonable level of maturity, it is checked into a software configuration control tool such as CVS.[7] Once that is done, any changes to the code must pass through a configuration control review board before they are incorporated. The modular nature of SST allows for different configurations with a variety of different components integrated together.

Simulation Procedure

The simulation must be designed, coded, and run through a verification, validation, and accreditation (VV&A) process[8]. Data will be presented in section 3.0 which will be used for validation. To capture this data, two reasonable scenarios were developed that would exercise key aspects of the simulation such as charging in and out of eclipse and orientation to the sun. The SST architecture allows any variable in the simulation to be recorded at each time step throughout the simulation. The lines of information in the data file are indexed by time into the simulation and by satellite number. This allows for data to be captured and analyzed for each individual satellite. The IPAC subsystem data is post-processed with tools such as spreadsheets and data parsers as required. The following section shows plots of key parameters such as power and communication capability as a function of time.

3.0 RESULTS AND ANALYSIS

The initial requirements of the test scenarios were to exercise how accurately the power model simulated the production and consumption of electricity as the satellite moved in and out of eclipse, and to drive the ADCS simulation. The same basic parameters were used in both a low earth orbit (LEO) and medium earth orbit (MEO) to compare the impact of the two orbits on subsystem performance.

Scenario

Two scenarios, each consisting of a single spacecraft at MEO or LEO, were utilized to generate data. Propagation of scenario orbits was set up in STK. The scenarios consisted of four days' worth of data, and included not only ephemeris data, but also attitude and eclipse information and payload activity. A slew profile was developed entailing several passes in earth-pointing, including payload operation, followed by several passes of solar inertial pointing. To simplify creation of the scenario, this profile was repeated for each day of the four days of ephemeris data generated.

LEO Scenario

Orbit elements for the LEO scenario were as follows: semi major axis, 7378.1 km (1000 km altitude, 105 minute period); inclination, 52 degrees; right ascension of the ascending node, 110 degrees; argument of perigee, 4.045 degrees. The orbit was circular. The LEO scenario data was generated at a 5 second time step, and includes periods of eclipse and several periods of payload activity.

MEO Scenario

Orbit elements for the MEO scenario were as follows: semi major axis, 16378.1 km (10,000 km altitude, 347.7 min period); argument of perigee, 358.1 degrees. The remaining orbit elements were identical to the LEO case and the orbit was again circular. The MEO scenario data was generated at a 1 minute time step.

Due to the inclination and the high altitude of the MEO scenario orbit, there are no eclipses. The payload is turned on several times in the MEO scenario, but the payload activity is more spread out (the same payload activity spread over a series of orbits takes a longer amount of time in the MEO scenario due to the longer period).

Interpreted Data

Attitude Model

Attitude data was verified by comparison to Satellite Tool Kit. Quaternion data is shown in figure 2 below. For the test case presented, the satellite has an altitude of 1134.9 km (circular orbit), and its z-axis is fixed to a ground located at 126° longitude and 39° latitude. The satellite's x-axis is fixed to the orbit plane. The comparison of data between STK and the ADCS module is acceptable: quantitatively, the magnitude of the RMS error vector between the two quaternions was .0076. For a circular orbit at 35786.0 km, the magnitude of the RMS error vector was .0015.

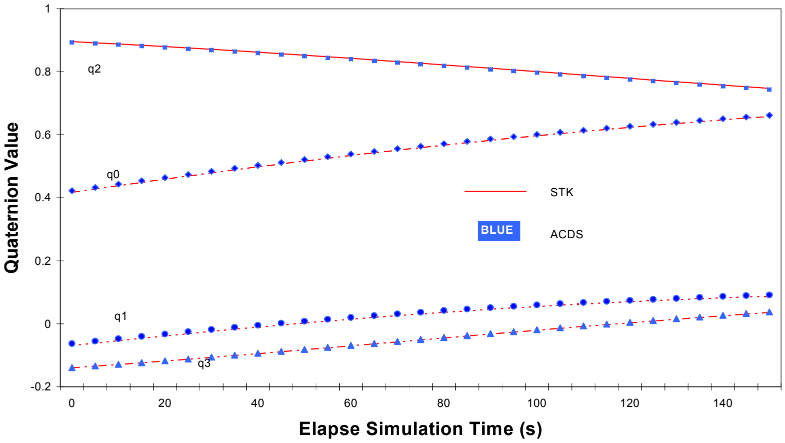


Figure 2: Graphical comparison of STK and ISBR/ADCS attitude quaternions for a LEO satellite pointing to a target.

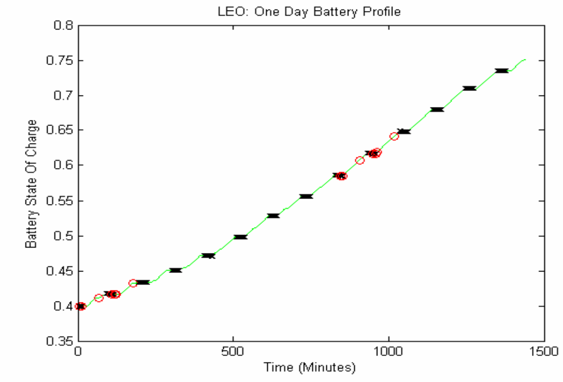
Eclipse data was also verified using STK. Eclipse is calculated by evaluating a distance variable, *d*:

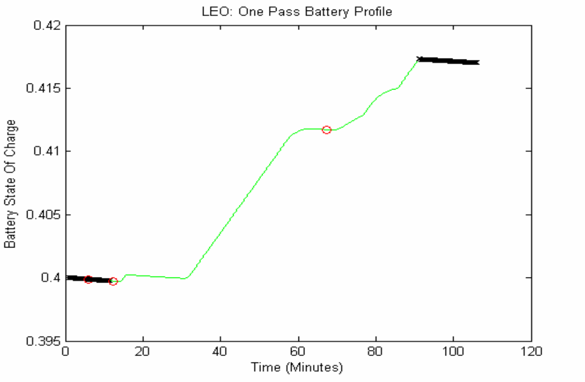
$$d = (R)(1-(e_{Sun}\cdot e_{Sat})^2)^{1/2}$$

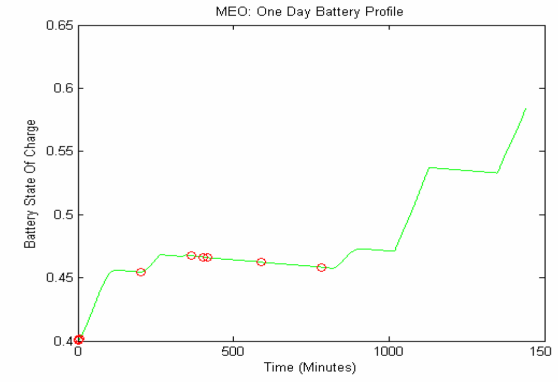
If $\boldsymbol{e_{Sun}}$ and $\boldsymbol{e_{Sat}}$ are on opposite sides of the Earth, then *d* is greater than the radius of the earth, and the satellite is in eclipse.

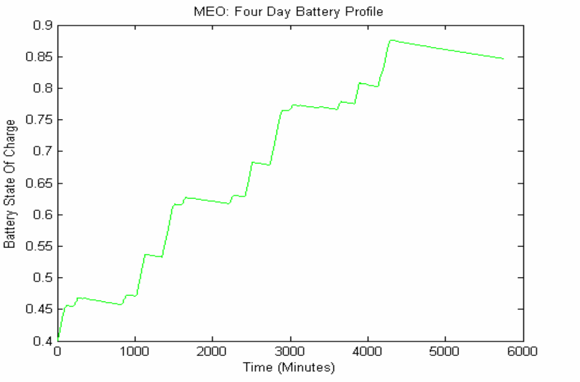
Power Subsystem Model

Data from the power model depicts an overall charging of the battery in both scenarios. In the LEO scenario, the battery starts at a 40% state of charge, and rises to approximately 75% over the first


LEO: One Day Battery Profile
Battery State Of Charge
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0
500
1000
1500
Time (Minutes)


LEO: One Pass Battery Profile
Battery State Of Charge
0.42
0.415
0.41
0.405
0.4
0.395
0
20
40
60
80
100
120
Time (Minutes)


MEO: One Day Battery Profile
Battery State Of Charge
0.65
0.6
0.55
0.5
0.45
0.4
0
500
1000
150
Time (Minutes)


MEO: Four Day Battery Profile
Battery State Of Charge
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0
1000
2000
3000
4000
5000
6000
Time (Minutes)

Position in orbit has the greatest effect on the communication (comm.) model, since this affects not only line-of-sight to a ground station, but also the range to any given ground station, which has a direct impact on data rate. Data was generated for the comm model using the MEO scenario, with four ground stations of interest selected: Schriever AFB, CO; Diego Garcia; New Boston, NH; and Thule, Greenland. Figure 5 shows the variation in data rate as the spacecraft passes within sight of a station. As contact is made with a station, there is a minimum data rate at approximately 150 kb/second; as the spacecraft passes, the rate increases as range decreases, and then declines. The minimum data rate is due to the fact that the spacecraft is always within a certain distance to the ground station if the station is in sight. Between stations, the data rate is zero.

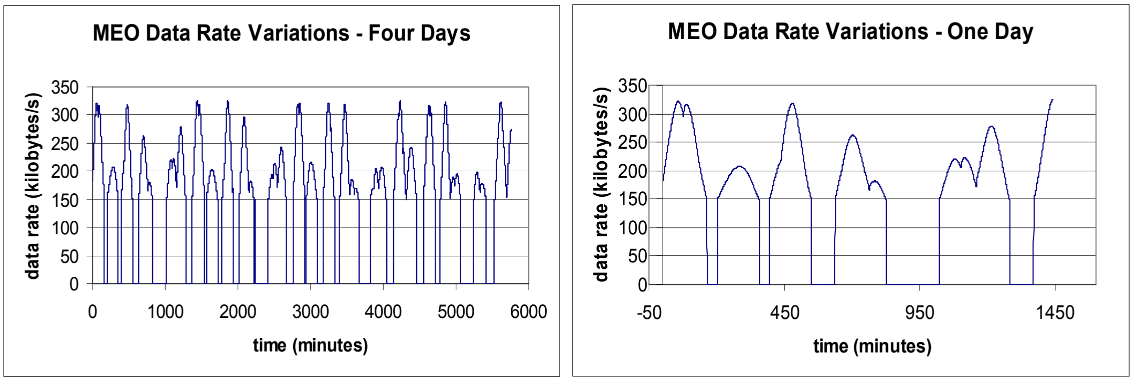


Figure 5: Four and One Day Data Rate Variations for MEO Scenario

Figure 5 shows four day and one day detail of the data rate variations, and gives a better view of how the data rates vary as a spacecraft passes a ground station. A “double hump” in the plot is due to a switch in ground station midstream as the data rate to another station becomes better than the current rate.

4.0 CHALLENGES OF END-TO-END SIMULATION

Among the more important lessons resulting from this research is the usability and timeliness of the simulation to the final customer. This includes getting requirements early enough in development, managing the customers’ expectations, and providing the simulation soon enough to be of use to the customer. In an ideal world of software development, the simulation systems engineer would start with a concept of what they want the tool to do. They would then survey customers and derive a thorough set of requirements for the simulation. Software developers, domain experts, program managers, etc... work with the systems engineer to identify an architecture and simulation sequence. These are then captured in diagrams such as UML Use Cases, Sequence Diagrams, Deployment Diagrams[9] and other such documents.

It is also critical to set realistic milestones and track the expenditures carefully. Software, in particular, has a history of cost and schedule over-runs. Careful project management and close attention to earned value are, albeit obvious, critical to successful software acquisition.

Typically when gathering requirements for simulations, the response from customers is very loosely defined. The project manager must then infer what the real requirements will be. This is best controlled by engaging the customer early with a notional set of requirements. Providing these notional requirements is a very effective way to draw the customer into the project. It also provides good opportunities to manage customer expectations early in development.

5.0 APPLICATIONS

There are multiple uses for a simulation of this sort. On the most general level, the tool can be used to examine the fundamental parameters of a mission. The question of whether a single spacecraft or a formation should be used may be aided by simulating how well tracking and orientation requirements are met using each option. Proposed orbits may be adjusted up or down, elliptical or circular in the same way. The utility of a mission as proposed can be assessed using the tool, and it can aid in the selection of alternate missions or revamped mission designs if necessary.

The simulation may be used for spacecraft design planning as well. A user can easily see over the course of a simulation, the benefit provided by a larger solar array or a better actuator. The outcome of tradeoffs, such as better power performance over several hours of simulated power system data, can be clearly seen in a manner which is not always possible with traditional analysis.

This capability allows for the planning of several passes as a whole, so that use of spacecraft resources can be optimized. Through the use of the SST, and creative subsystem model development, the cost and risk of flying satellites can be greatly mitigated.

6.0 REFERENCES

[1] De Jonckheere, R.K., et al., “Micro-satellite Modeling, Simulation, And Analysis in the Distributed Architecture Simulation Laboratory,” AIAA, 2001-4742, 1999.

[2] *Ibid.*

[3] Stargardt, B., Schum, K., “A Modeling, Simulation and Analysis Architecture for Space Technology Assessment,” Core Technologies in Space Conference, Colorado Springs, CO, 2003

[4] Nise, Norman S., Control Systems Engineering, John Wiley & Sons, 2000.

[5] Wertz, J., Larson, W., Space Mission Analysis and Design, 3ed, Microcosm Press, 2003.

[6] *Ibid.*

[7] http://www.nongnu.org/cvs/, 2006

[8] Cockburn, A., Writing Effective Use Cases, 4ed, Addison Wesley, 2001

[9] Rosenberg, D., Scott, K., Use Case Driven Object Modeling with UML, a Practical Approach, 1999